


Q5

--The write operation and write barrier process are described with reference to Figure 5. In Step 5.1 a write operation is called and intercepted by the write barrier which is integrated with the write operation code. Step 5.2 checks whether the object is being assigned to a static variable of a class and if so set the status of the object as global (step 5.4) and set all objects reachable by the object as global (step 5.5) before moving to step 5.6, if not so then do step 5.3. Step 5.3 checks whether the write operation assigns the object to another global object, if so set the status of the object as global (step 5.4) and set all objects reachable by the object as global (step 5.5). Once complete continue with the write operation.--

In the claims:

- Sub BI
1. (Amended) A method of managing memory in a multi-threaded processing environment including local thread stacks and local thread heaps, and a global heap, said method comprising:
creating an object in a thread heap; and
for a given thread, monitoring each object that is local to the given thread, to determine whether the object is accessible from any thread other than the given thread.
 - Q6
 2. (Amended) The method as claimed in claim 1 further comprising:
assigning a status to the given object, the status designating the object as a local object.
 3. (Amended) The method as claimed in claim 2 further comprising deleting from the thread heap one or more local objects when it is determined that they are not accessible from a local root.

- 
4. (Amended) The method as claimed in claim 2, further comprising changing the status of the object to global when the monitoring step determines that the object is accessible from either of a global root or other global object.
 5. (Amended) The method as claimed in claim 2, further comprising changing the status of an object in the thread heap to global if the object is assigned to a static variable or if the object is assigned to a field in a global object.
 6. (Amended) The method as claimed in claim 3 further comprising intercepting assignment operations to an object in the thread heap to determine whether the object status should be changed.
 7. (Amended) The method as claimed in claim 6 wherein the multithreaded processing environment is a virtual machine.
 8. (Amended) The method as claimed in claim 7 wherein the virtual machine comprises an interpreter comprising a write operation code modified to perform a checking of assignment of the object.
 9. (Amended) The method as claimed in claim 8 wherein the virtual machine comprises a just in time compiler wherein native compiled write operation code includes native code to perform the checking of assignment of the object.
 10. (Amended) The method as claimed in claim 9 further comprising using spare capacity in an object header for the status.

11. (Amended) The method as claimed in claim 10 further comprising using multiples of 2 or more bytes in a thread heap to store the objects whereby there is at least one spare bit in the object length variable and using the at least one spare bit as the status.

12. (Amended) The method as claimed in claim 11 further comprising moving objects whose status is global from the thread heap to the global heap.

13. (Amended) The method as claimed in claim 12 further comprising compacting the reachable local objects in a thread heap.

14. (Amended) The method as claimed in claim 1 further comprising assigning a status to certain objects that designates the objects as local objects.

15. (Amended) The method as claimed in claim 14 where said certain objects include class objects.

16. (Amended) The method as claimed in claim 14 further comprising the step of analysing whether an object is likely to be made global and associating such an object with a global status on creation.

17. (Amended) The method as claimed in claim 16 further comprising allocating objects assigned as global on creation to the global heap.

18. (Amended) A system for managing memory in a multi-threaded processing environment comprising:

respective local thread stacks and heaps;
a global heap;
means for creating an object in a thread heap; and

means for monitoring each object that is local to the given thread, to determine whether the object is accessible from any thread other than the given thread.

19. (Amended) The system as claimed in claim 18 further comprising means for associating a local status with the object and means for changing the status of the object to global under certain conditions.

20. (Amended) The system as claimed in claim 19 further comprising means for deleting from the thread heap one or more local objects when they are not accessible from a local root.

21. (Amended) The system as claimed in claim 20 further comprising:
means for changing the status of an object in the thread heap to global if the object is assigned to a static variable or if the object is assigned to a field in a global object.

22. (Amended) A computer program product stored on a computer readable storage medium for, when executed on a computer, performing a method of managing memory in a multi-threaded processing environment including local thread stacks and local thread heaps, and a global heap, said method comprising:

creating an object in a thread heap; and
for a given thread, monitoring each object that is local to the thread, to determine whether the object is accessible from any thread other than the given thread.

23. (Amended) The computer program product as claimed in claim 22 further comprising:
assigning a status with the object that designates the object as a local object.

24. (Amended) The computer program product as claimed in claim 23 further comprising deleting from the thread heap one or more local objects when it is determined that they are not reachable from a local root.